REMARKS

The rejections presented in the Office Action dated November 13, 2003 have been considered.

Claim 1 is amended to include the limitations of now-canceled claim 2 for purposes of expediting prosecution, and claim 15 is similarly amended. Claim 2 is canceled without prejudice. Claim 3 is amended to change the dependency. New claims 16 and 17 are added to further clarify the invention.

A Supplemental IDS was filed January 16, 2004 with copies of the documents cited in the previously filed IDS.

Acceptance of the drawings by the Examiner is acknowledged.

The Examiner's amendment to the specification is acknowledged and accepted.

The Examiner's interpretations of the identified terms are not unacceptable for purposes of examination.

The claims are thought to be allowable over the cited art as explained in the arguments set forth below.

The Office Action fails to establish that claims 1-15 are anticipated by US patent number 6,189,141 to Benitez et al. ("Benitez") under 35 USC §102(e). The rejection is traversed because the Office Action fails to show that all the limitations of the claims are identically taught by Benitez.

The rejection of claim 1 is traversed. However, the rejection is now moot in view of the limitations of claim 2 that have been added to claim 1. The rejection also fails to show that Benitez teaches the limitations of now-canceled claim 2. The limitations of combined claims 1 and 2 include patching the function entry points with breakpoint instructions; creating a shared memory segment for the instrumentation program and the application program; upon initial invocation of the original functions in the application program and in response to encountering the breakpoint instructions, creating in the shared memory segment corresponding substitute functions including instrumentation code; and executing the substitute functions in lieu of the original functions in the application program.

The Office Action alleges that Benitez's col. 2, ll. 53-55 teaches patching the function entry points with breakpoint instructions and creating substitute functions upon encountering the breakpoint instructions. However, this cited section of Benitez teaches, "A trace typically is made up of one or more blocks of original instructions of an executable file, each of which

may be reached through a common control path." There is no apparent suggestion of patching with breakpoint instructions, nor is there any apparent suggestion of triggering creation of substitute functions by encountering the breakpoint instructions. Furthermore, it appears that Benitez first identifies a hot block based on the number passes through the starting instruction, which causes the hot trace selector 204 to select a hot trace including the hot block (col. 9, ll. 45-59). Benitez's FIG. 9 shows an intermediate representation generator 910 generating hot trace IR 912 in response to the hot trace selector 204. Thus, Benitez does not appear to teach patching with break instructions and creating the substitute functions when the break instructions are encountered. Claim 1, as amended with the limitations of now-canceled claim 2, is not shown to be anticipated by Benitez, and withdrawal of the rejection is respectfully requested.

As to claim 3, the Office Action fails to show that Benitez identically teaches the limitations of replacing the break instruction at the entry points of the functions in the application program with branch instructions that target the substitute functions. In combination with claim 1, the claimed process first patches with a break instruction and then after creating the substitute functions replaces the break instruction with a branch instruction. The cited portion of Benitez (col. 2, ll. 60-65) states, "A basic block has at least one exit instruction from which control passes out of the basic block to another block. A control path from one block to another block is referred to herein as an arc. The action of transferring control over an arc, other than by an unconditional fall-through is referred to as a jump." This does not appear to be related to patching with break instructions and then replacing the break instructions with branches. Furthermore, no other teaching of Benitez appeared to suggest the claimed approach. Therefore, the Office Action fails to show that Benitez teaches the limitations of claim 3, and the rejection should be withdrawn.

Claim 4 depends from claim 3 and is not anticipated for at least the reasons set forth above.

Claim 5 depends from claim 1 and is not anticipated for at least the reasons set forth above.

Claim 6 depends from claim 1 and includes further limitations of copying a segment of the executable application program to selected area of memory by the instrumentation program; replacing the segment of the application program with code that allocates the shared memory by the instrumentation program; executing the code in the application program that allocates the shared memory segment; and restoring the segment of the executable application

from the selected area of memory to the application program by the instrumentation program after the shared memory is allocated. The Office Action fails to show that these limitations are identically taught by Benitez in the cited Benitez's col. 12, ll. 20-50 and col. 34's description of the Hot Trace Memory Manager.

Benitez's col. 12, ll. 20-50 discusses correlating original instructions with translated instructions in a lookup table and using the lookup table to determine whether to transfer control to the translated instruction. Benitez's description of the Hot Trace Memory Manager in col. 34 indicates that this component stores optimized and instrumented hot trace into hot trace storage and selectively removes from the storage area those hot blocks that have become cold. These sections do not suggest the limitations of claim 6.

Claim 6 clearly indicates that the application program (not an instrumentation program or Hot Trace Memory Manager) allocates the shared memory used for instrumentation by way of a segment of the application program that is temporarily replaced with code that allocates memory. This temporary replacement of application program code with code that allocates the shared memory segment is not taught by the cited sections of Benitez, nor seen to be suggested elsewhere in Benitez. Therefore, the Office Action fails to show that Benitez teaches the limitations of claim 6, and the rejection should be withdrawn.

Claims 7-14 depend directly or indirectly from the claims discussed above, and therefore, are allowable for at least the reasons set forth above.

Claim 15 is amended and includes limitations similar to those of amended claim 1. Therefore, claim 15 is not anticipated by Benitez.

Withdrawal of the rejection and reconsideration of the claims are respectfully requested in view of the remarks set forth above.

Respectfully submitted,

CRAWFORD MAUNU PLLC
1270 Northland Drive, Suite 390
Saint Paul, MN 55120
(651) 686-6633

By: _____

Name: LeRoy D. Maunu
Reg. No.: 35,274